1. (30 points) *PCA via successive deflation.* Let $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k$ be the eigenvectors of the k largest eigenvalues of $\boldsymbol{C} = (1/n)\boldsymbol{\Phi}\boldsymbol{\Phi}^\top$, i.e., the PCA solution. These satisfy

$$\boldsymbol{\theta}_c^\top \boldsymbol{\theta}_d = \begin{cases} 0 & c \neq d \\ 1 & c = d. \end{cases}$$

We will construct a method of finding $\boldsymbol{\theta}_c$ sequentially.

As we showed in class, $\boldsymbol{\theta}_1$ is the first principal eigenvector of $\boldsymbol{C}$, and satisfies $\boldsymbol{C}\boldsymbol{\theta}_1 = \lambda_1 \boldsymbol{\theta}_1$. Now define $\widetilde{\boldsymbol{\phi}}_i$ as the orthogonal projection of $\boldsymbol{\phi}_i$ onto the space orthogonal to $\boldsymbol{\theta}_1$:

$$\widetilde{\boldsymbol{\phi}}_i = (\boldsymbol{I} - \boldsymbol{\theta}_1 \boldsymbol{\theta}_1^\top)\boldsymbol{\phi}_i.$$

Define $\widetilde{\boldsymbol{\Phi}} = [\ \boldsymbol{\phi}_1 \ \cdots \ \boldsymbol{\phi}_n\ ]$ as the *deflated* data matrix, we have

$$\widetilde{\boldsymbol{\Phi}} = (\boldsymbol{I} - \boldsymbol{\theta}_1 \boldsymbol{\theta}_1^\top)\boldsymbol{\Phi}.$$

(a) Using the fact that $\boldsymbol{C}\boldsymbol{\theta}_1 = \lambda_1 \boldsymbol{\theta}_1$ and $\|\boldsymbol{\theta}_1\|^2 = 1$, show that the second moment of the deflated matrix is given by

$$\widetilde{\boldsymbol{C}} = \frac{1}{n}\widetilde{\boldsymbol{\Phi}}\widetilde{\boldsymbol{\Phi}}^\top = \boldsymbol{C} - \lambda_1 \boldsymbol{\theta}_1 \boldsymbol{\theta}_1^\top.$$

(b) Show that $\boldsymbol{\theta}_2$ is the eigenvector of the largest eigenvalue of $\widetilde{\boldsymbol{C}}$. Recall that $\boldsymbol{\theta}_2$ is the eigenvector of the second-largest eigenvalue of $\boldsymbol{C}$.

(c) Suppose we have a simple method for finding the leading eigenvector and eigenvalue of a positive semi-definite matrix, denoted by $(\lambda, \boldsymbol{u}) = f(\boldsymbol{C})$. Write some pseudo code for finding the first k principal basis vectors of $\boldsymbol{\Phi}$ that only uses the special f function and simple vector arithmetic, i.e., your code should not use SVD or the eig function. Hint: this should be a simple iterative routine that takes 2–3 lines to write. The input is $\boldsymbol{C}$, k and the function f, the output should be $\boldsymbol{\theta}_c$ and λ_c for $c = 1, \ldots, k$. Do not worry about being syntactically correct.

Ans.1 ⇒

(a) = given that ⇒

$$\theta_c^T \theta_d = \begin{cases} 0 & c \neq d \\ 1 & c = d \end{cases}$$

So we know $\boxed{\theta_1^T \theta_1 = 1}$ $(c = d = 1)$

↳ (1)

also given that ⇒

$$\widetilde{\phi} = (I - \theta_1 \theta_1^T)\phi \longrightarrow (2)$$

from matrix properties we know that ⇒

$$(XY)^T = Y^T X^T$$

and $(X)^T = X$ (if X is symmetric)

Here we know that ⇒

$(I - \theta_1 \theta_1^T)$ = Symmetric matrix

Hence $\boxed{(I - \theta_1 \theta_1^T)^T = (I - \theta_1 \theta_1^T)}$ → (3)

To prove ⇒

$$\tilde{C} = \frac{1}{n}\tilde{\phi}\tilde{\phi}^T = C - \lambda_1 \theta_1 \theta_1^T$$

So,

$$\tilde{C} = \frac{1}{n}\tilde{\phi}\,\tilde{\phi}^T$$

Putting value of $\tilde{\phi}$ from equation ② ⇒

$$\tilde{C} = \frac{1}{n}\left[\left(I - \theta_1\theta_1^T\right)\phi \left(\left(I - \theta_1\theta_1^T\right)\phi\right)^T\right]$$

using fact $(XY)^T = Y^T X^T$, $(I - \theta_1\theta_1^T)$ is symmetric

$$\tilde{C} = \frac{1}{n}\left[\left(I - \theta_1\theta_1^T\right)\phi\,\phi^T\left(I - \theta_1\theta_1^T\right)\right]$$

$$\tilde{C} = \frac{1}{n}\left[\left(I\cdot\phi\phi^T - \theta_1\theta_1^T\phi\,\phi^T\right)\left(I - \theta_1\theta_1^T\right)\right]$$

$$\tilde{C} = \frac{1}{n}\left[\left(\phi\phi^T - \theta_1\theta_1^T\phi\,\phi^T - \phi\phi^T\theta_1\theta_1^T + \theta_1\theta_1^T\phi\phi^T\theta_1\theta_1^T\right)\right]$$

given that: $C\theta_1 = \lambda_1\theta_1$

$$\left(\frac{1}{n}\phi\phi^T\right)\theta_1 = \lambda_1\theta_1$$

$$\boxed{\phi\phi^T\theta_1 = n\lambda_1\theta_1} \rightarrow (4)$$

and $$(\phi\phi^T\theta_1)^T = (n\lambda_1\theta_1)^T$$

$$\boxed{\theta_1^T\phi\phi^T = n\lambda_1\theta_1^T} \rightarrow (5)$$

Putting values from (4) and (5) we get:

$$\tilde{C} = \frac{1}{n}\left[\phi\phi^T - \theta_1 n\lambda_1\theta_1^T - n\lambda_1\theta_1\theta_1^T + \theta_1 n\lambda_1\theta_1^T\theta_1\theta_1^T\right]$$

Since $[\theta_1^T\theta_1 = 1 \Rightarrow$ from equation 0$]$

$$\tilde{C} = \frac{1}{n}\left[\phi\phi^T - \theta_1 n\lambda_1\theta_1^T - n\lambda_1\theta_1\theta_1^T + \theta_1 n\lambda_1 \cdot 1 \cdot \theta_1^T\right]$$

$$\tilde{C} = \frac{1}{n}\left[\phi\phi^T - \cancel{\theta_1 n\lambda_1\theta_1^T} - n\lambda_1\theta_1\theta_1^T + \cancel{\theta_1 n\lambda_1\theta_1^T}\right]$$

$$\tilde{\tilde{C}} = \frac{1}{n}\left[\phi\phi^T - n\lambda_1 v_1 v_1^T\right]$$

$$\tilde{C} = \frac{1}{n}\phi\phi^T - \lambda_1 v_1 v_1^T$$

Hence proved

$$\tilde{\tilde{C}} = C - \lambda_1 v_1 v_1^T$$

Hence we can say that =

$$\tilde{C} = \frac{1}{n}\tilde{\phi}\tilde{\phi}^T = C - \lambda_1 v_1 v_1^T$$

(b) = we have proved in part (1) =

$$\tilde{C} = \left(\frac{1}{n}\phi\phi^T - \lambda_1\theta_1\theta_1^T\right)$$

multiplying both sides by θ_j =

$$\tilde{C}\theta_j = \left(\frac{1}{n}\phi\phi^T - \lambda_1\theta_1\theta_1^T\right)\theta_j$$

$$\tilde{C}\theta_j = \frac{1}{n}\phi\phi^T\theta_j - \lambda_1\theta_1\theta_1^T\theta_j$$

from part (1) we know $\left(\phi\phi^T\theta_j = n\lambda_1\theta_j\right)$

so we get,

$$\tilde{C}\theta_j = \lambda_j\theta_j - \lambda_1\theta_1\theta_1^T\theta_j$$

also $\left[\theta_1^T\theta_j = 0 \text{ , hence } \lambda_1\theta_1\theta_1^T\theta_j = 0\right)$ $\forall j \neq 1$

Hence,

(when $j \neq 1$)

$$\boxed{\tilde{C}\theta_j = \lambda_j\theta_j}$$

when $(J=1)$

$$\tilde{C}\theta_1 = \lambda_1\theta_1 - \lambda_1\theta_1\theta_1^T\theta_1 \qquad [\theta_1^T\theta_1 = 1]$$

$$\tilde{C}\theta_1 = \lambda_1\theta_1 - \lambda_1\theta_1$$

$$\tilde{C}\theta_1 = (\lambda_1 - \lambda_1)\theta_1$$

$$\tilde{C}\theta_1 = (0)\theta_1$$

$$\boxed{\tilde{C}\theta_1 = 0}$$

Hence for $(J \neq 1)$ θ_J is a principle eigenvector of $\tilde{C}$ with same eigenvalue λ_J. Also θ_1 is an eigenvector of $\tilde{C}$ with eigenvalue 0.

Since $\theta_1, \theta_2 \dots \theta_k$ are the first k eigenvectors with largest eigenvalues of C i.e the principal basis vectors therefore

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_k$$

we know that for $\tilde{C}$, θ_j are the Principal eigenvectors with eigenvalues $(\theta_1, \lambda_1, \lambda_2 \text{ ---- } \lambda_k)$ Therefore from above equation we can say that λ_2 is the largest eigenvalue of $\tilde{C}$ (since λ_1 is not an eigenvalue of $\tilde{C}$)

Hence θ_2 is the first Principle eigenvector.

(c) = Pseudocode to find first k principal basis vectors of ϕ.

```
function obtainPrincipalBasisVectors(C, K, f):
    list_λ = []
    list_u = []
    for loop i in range(K):
        λ, u = f(C)
        C = C - λ * u * u.Transpose
        (list_λ).append(λ)
        (list_u).append(u)
    return (list_u), (list_λ)
```

2. (20 points) Consider a Gaussian mixture model in which the marginal distribution of the latent variable $\boldsymbol{y}$ is $\Pr(\boldsymbol{y} = \boldsymbol{e}_c) = \pi_c, c = 1, \ldots, k$, and the conditional distribution of $\boldsymbol{x}$ given $\boldsymbol{y}$ is $\mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma})$, i.e., each Gaussian component has their own mean $\boldsymbol{\mu}_c$ but they share the same covariance matrix $\boldsymbol{\Sigma}$. Given a set of observations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, derive the expectation-maximization algorithm for estimating the model parameters $\pi_c, \boldsymbol{\mu}_c, c = 1, \ldots, k$, and $\boldsymbol{\Sigma}$.

Ans.2 =

Expectation - Maximization Algorithm =

The Expectation - maximization algorithm in an iterative method to find the M.L.E. or M.A.P. estimate for models with latent variables.

Steps for EM algorithm =

1) = Initialization = first we randomly initialize parameters

θ^0 (all the μ_c, Σ and π_c values)

2) = Initialize (mu, sigma, pi) with random values.

mu = [rand()] * k

sigma [rand()] * k

pi = [rand()] * k

[k = number of gaussians]

2- Expectation Step = By Assuming that parameter (θ^{t-1}) values from Previous step are fixed we compute the expected values of the latent variable. (or a function of the expected values of latent variables)

calculating Expectation (ψ_{ic}) ⇒

$$\psi_{ic} = \frac{\pi_c \, N(x_i | \mu_c, \Sigma)}{\sum_{j=1}^{k} \pi_j \, N(x_i | \mu_j, \Sigma)}$$

where x_i = observations.

(Σ, μ, π) = parameters values from step ①.

(3) = Maximization Step = from the given values we computed in last step (value of latent variables) we estimate new values (θ^t) that maximize a variant of likelihood function.

Re-estimate and update parameters using current ψ_{ic} from previous step.

$$\mu_c^{new} = \frac{1}{\sum_{i=1}^{n} \psi_{ic}} \left[\sum_{i=1}^{n} \psi_{ic} x_i \right]$$

$$\pi_c^{new} = \frac{\sum_{i=1}^{n} \psi_{ic}}{n}$$

$$\Sigma^{new} = \frac{\sum_{i=1}^{n} \sum_{c=1}^{K} \psi_{ic} (x_i - \mu_c)(x_i - \mu_c)^T}{\sum_{i=1}^{n} \sum_{c=1}^{K} \psi_{ic}}$$

④ = Exit condition = if likelihood of observations have not changed much then we stop the iterations otherwise we go back to step ② and Repeat steps ② and ③ until convergence.

Suppose $L(x, \mu_c, \Sigma, \pi_c)$ is a function to compute log-likelihoods.

Then we will calculate log likelihood for current parameters and previous parameters and compare them against a threshold (tolerated change for log likelihood)

- old_L = $L(x, \mu_c, \Sigma, \pi_c)$
- new_L = $L(x, \mu_c^{new}, \Sigma^{new}, \pi_c^{new})$
- if absolute (old_L − new_L) < tolerated threshold (ex = 0.001)
  - Then Exit
  - else
    - go back to step ②.

3. (50 points) *20 Newsgroup revisited.* Let us revisit the 20 Newsgroup data set <http://qwone.com/~jason/20Newsgroups/>, and apply some of the unsupervised methods by ignoring their labels. We will only consider the training data. You are required to code the algorithms by yourselves in the language of your choice.

   (a) *LSI/PCA via orthogonal iteration.* Implement the orthogonal iteration algorithm that finds the PCA projection matrix $\boldsymbol{\Theta}$ of a data matrix $\boldsymbol{\Phi}$. You are allowed to use a pre-existing function of QR. Apply tf-idf to the term-document matrix to obtain $\boldsymbol{\Phi}$ and feed it into your orthogonal iteration algorithm. Remember to use sparse matrix operations to avoid unneccessary memory/computational complexities. Set $k = 2$ and let the algorithm run until $\boldsymbol{\Theta}$ doesn't change much. Then get $\boldsymbol{Y} = \boldsymbol{\Theta}^\top \boldsymbol{\Phi}$. Each column of $\boldsymbol{Y}$ is a two-dimensional vector that you can plot on a plain. Plot all the documents on a two-dimensional plain, and use a different color for each point that belong to different news groups.

   (b) *GMM via EM.* Implement the EM algorithm for the Gaussian mixture model (with different means and covariances for each Gaussian component). The data matrix $\boldsymbol{\Phi}$ is obtained from LSI with $k_{\text{LSI}} = 100$ using the previous orthogonal iteration algorithm. Run the EM algorithmn for GMM with $k_{\text{GMM}} = 20$ until convergence. For each Gaussian component $\mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$, calculate $\boldsymbol{\Theta}\boldsymbol{\mu}_c$ where $\boldsymbol{\Theta}$ is the PCA projection; the vector $\boldsymbol{\Theta}\boldsymbol{\mu}_c$ should be element-wise nonnegative. For each cluster c, show the 10 terms that have the highest value in $\boldsymbol{\Theta}\boldsymbol{\mu}_c$. The index-term mapping can be found here <http://qwone.com/~jason/20Newsgroups/vocabulary.txt>. Does the result make sense?

Ans.3 =

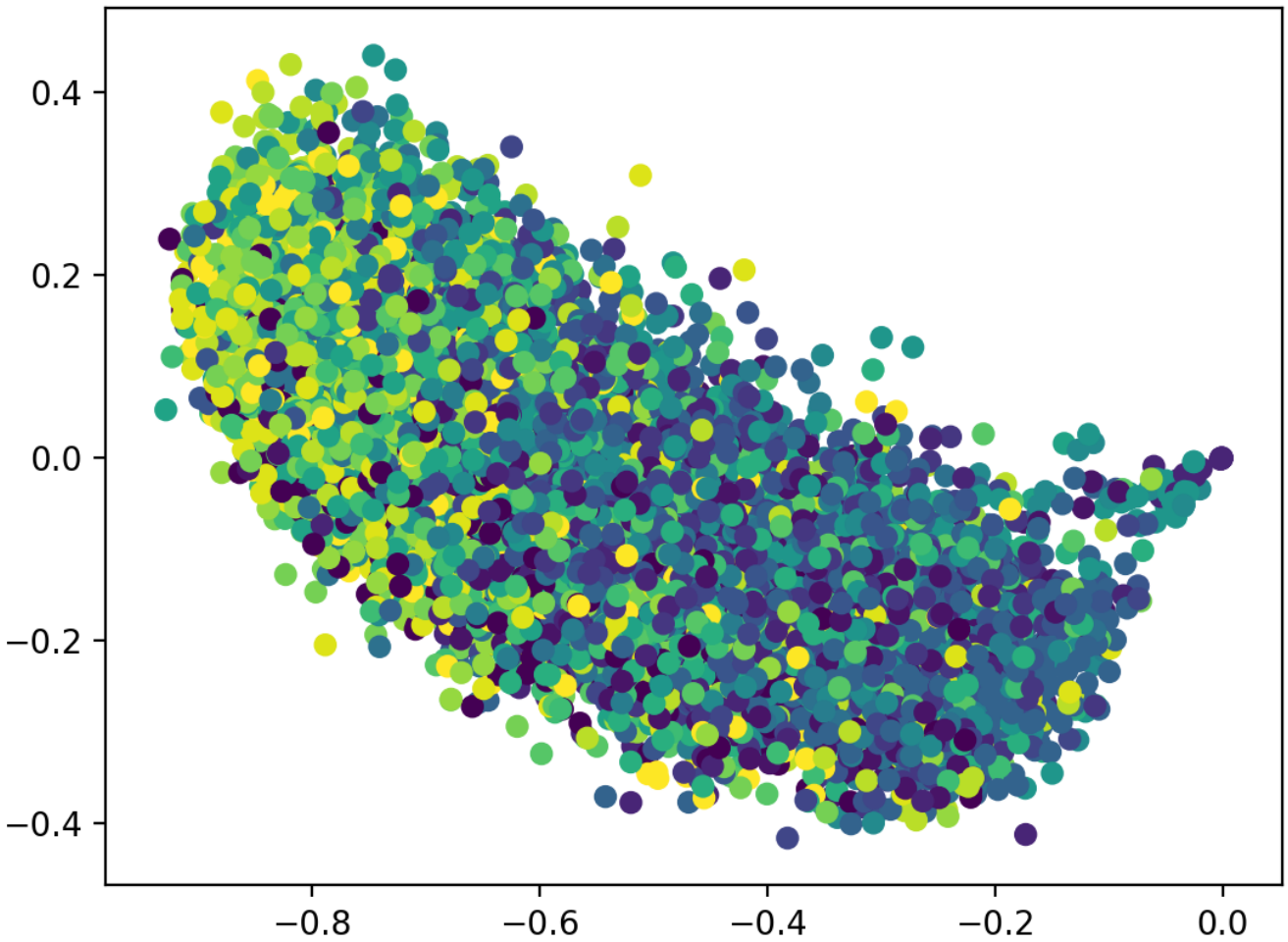
Part (a) =

The code for the program is included with the assignment submission.

Code file Name =

LSI_PCA_orthogonal_iteration.py

By running this program following plot for 20newsgroup Dataset is obtained =



Different intensities of green color represent 20 different clusters of newsgroups.

Part (b) =

The code file is added with the assignment.

This prints top ten terms with Highest value of (θ_{uc}) for each cluster.

Higher the value means the word is most commonly used in the articles of that newsgroup Cluster.